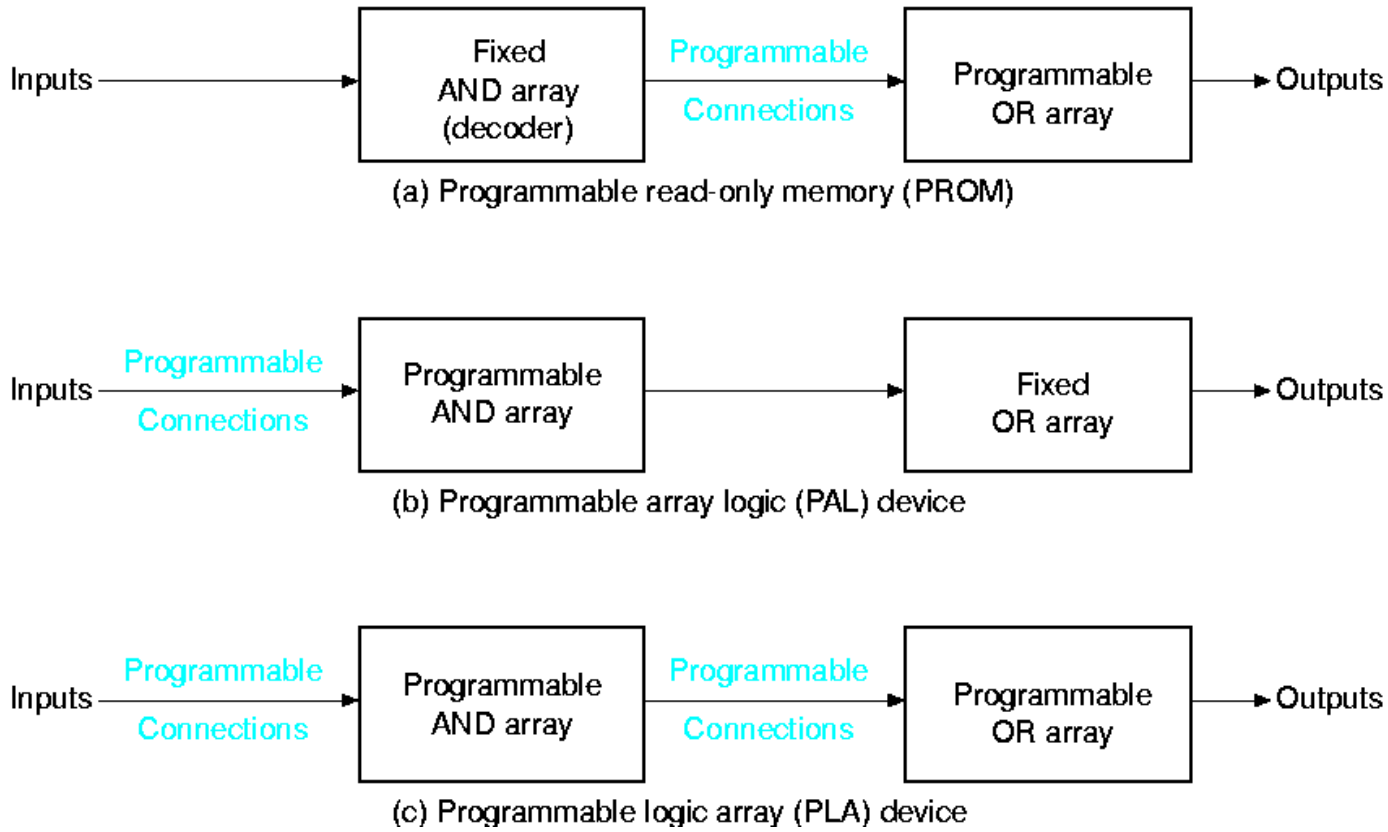
PROGRAMMABLE LOGIC DEVICES



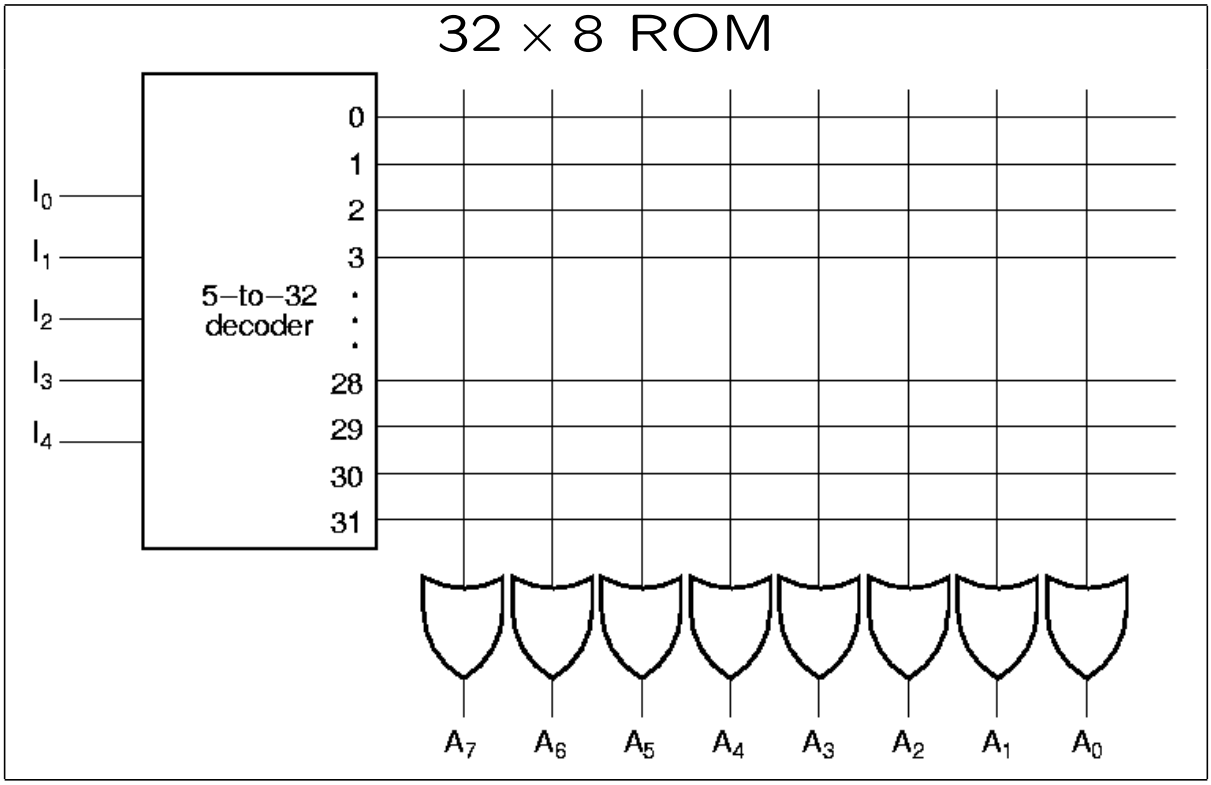
PLDs (combinatorial circuits): ROM, PLA, PAL, CPLD, and FPGA

Store *permanent* binary information (nonvolatile). Can be read only (cannot be altered). Information is specified by designer and *physically inserted* (embedded) into the PLD

Programmable connections are formed by *fuses*, *masks*, or *antifuses* depending on the technology. Irreversible programming

Read-Only Memory

k inputs (address) $\Rightarrow$ $\boxed{\begin{array}{c}2^k \times n \\ \text{ROM}\end{array}}$ $\Rightarrow$ n outputs (data)



- $k \times 2^k$ decoder to decode input address
- n OR gates with 2^k input each
- Decoder output is connected to all n OR gates through fuses
- ROM $\rightarrow 2^k \times n$ programmable connections

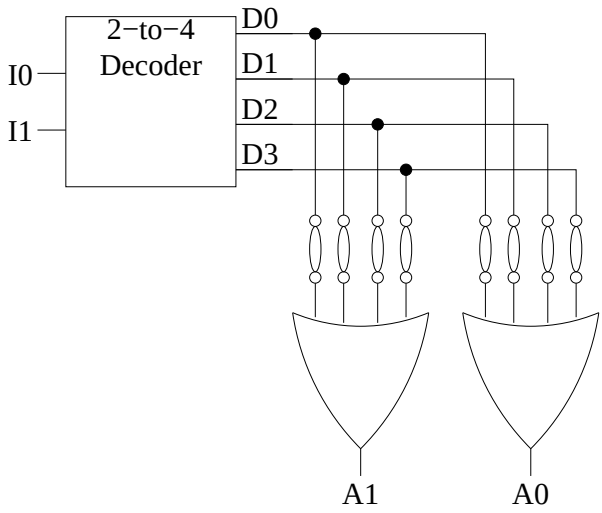
Programming a ROM

Example of 4×2 ROM

Truth table

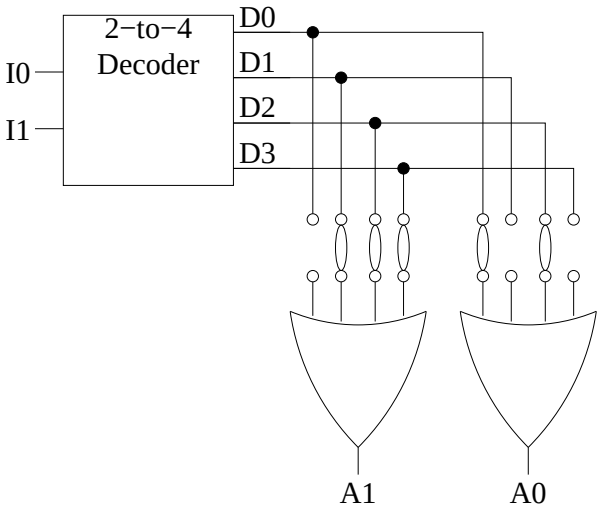
Address		Content	
I_1	I_0	A_1	A_0
0	0	0	1
0	1	1	0
1	0	1	1
1	1	1	0

NonProgrammed ROM

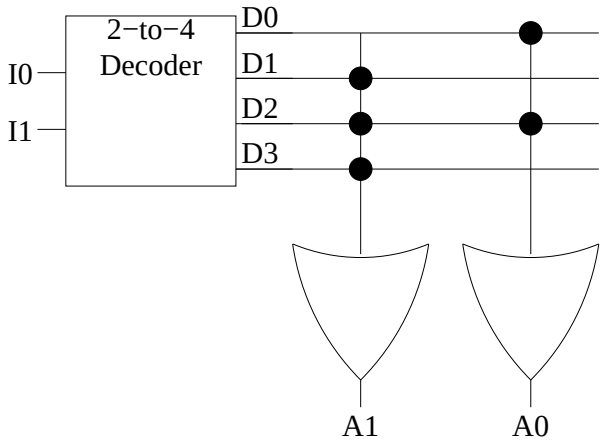


⇓

Programmed ROM



Compact ROM



Truth table → address and content of ROM

Programming → stores truth table in ROM

- 0 = Open connection = Fuse blown
- 1 = Closed connection = Fuse intact

Function Synthesis with ROM

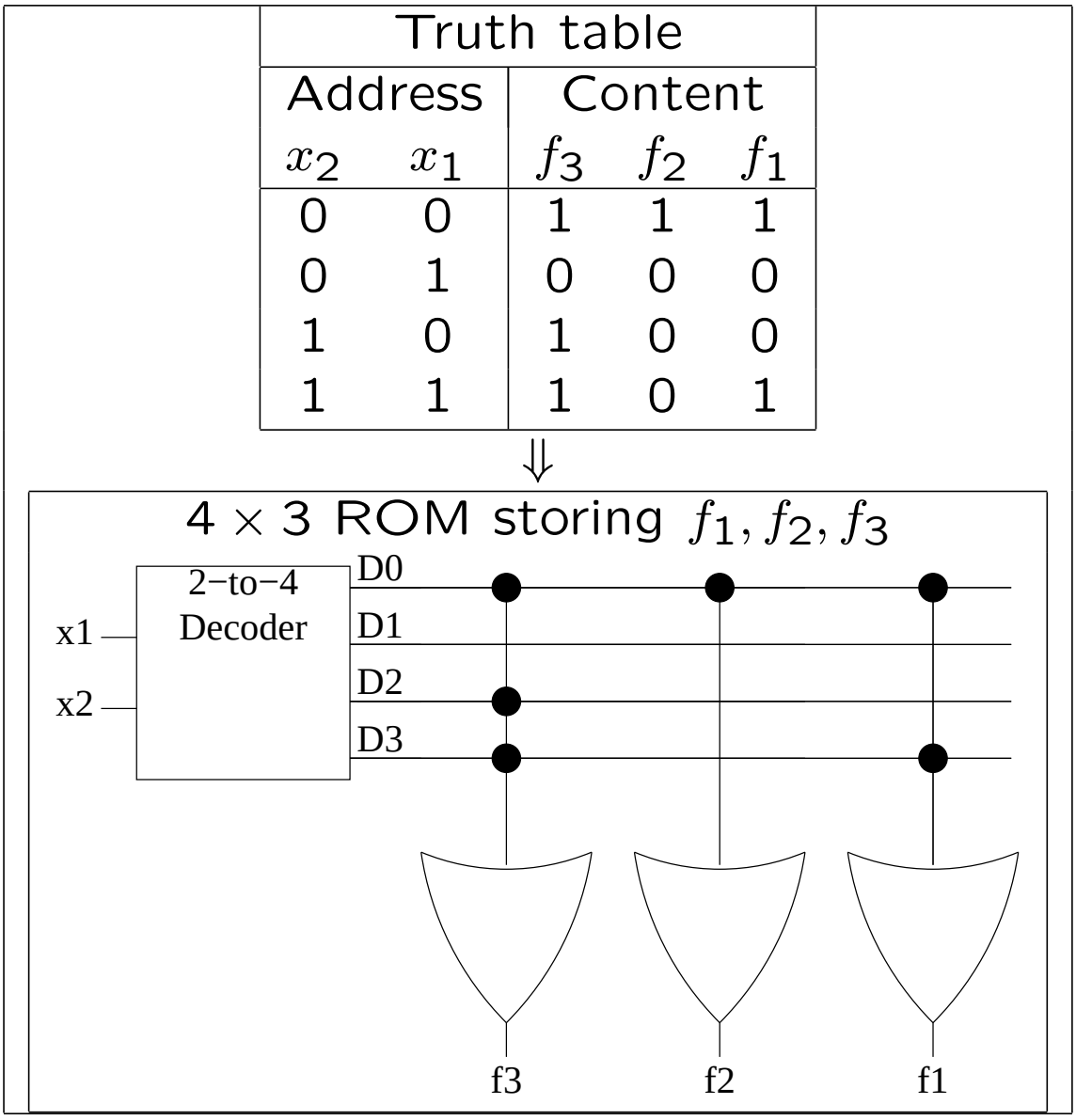
Any set of functions $f_1(x_k, \ldots, x_1)$, ..., $f_n(x_k, \ldots, x_1)$ can be realized with a $2^k \times n$ ROM

Example: Implement $f_1(x_2, x_1) = \sum m(0, 3)$, $f_2(x_2, x_1) = \overline{x_2 + x_1}$, and $f_3(x_2, x_1) = \prod M(1)$ with a 4×3 ROM

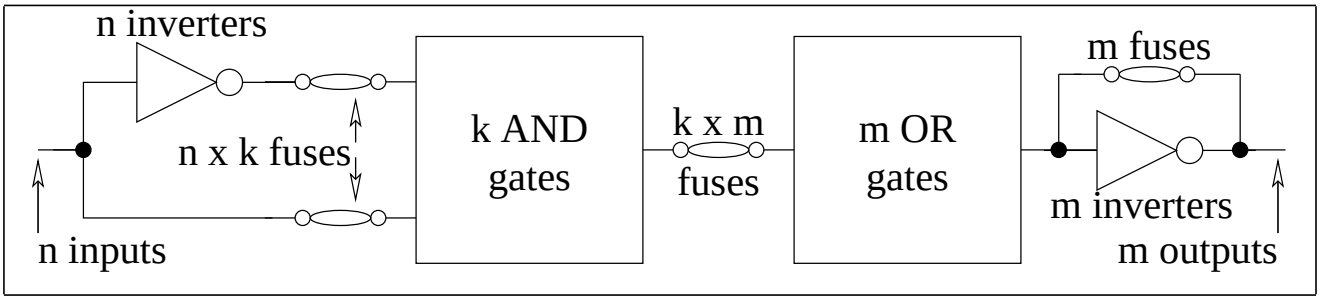
Truth table

Address		Content		
x_2	x_1	f_3	f_2	f_1
0	0	1	1	1
0	1	0	0	0
1	0	1	0	0
1	1	1	0	1

$\Downarrow$



Programmable Logic Array



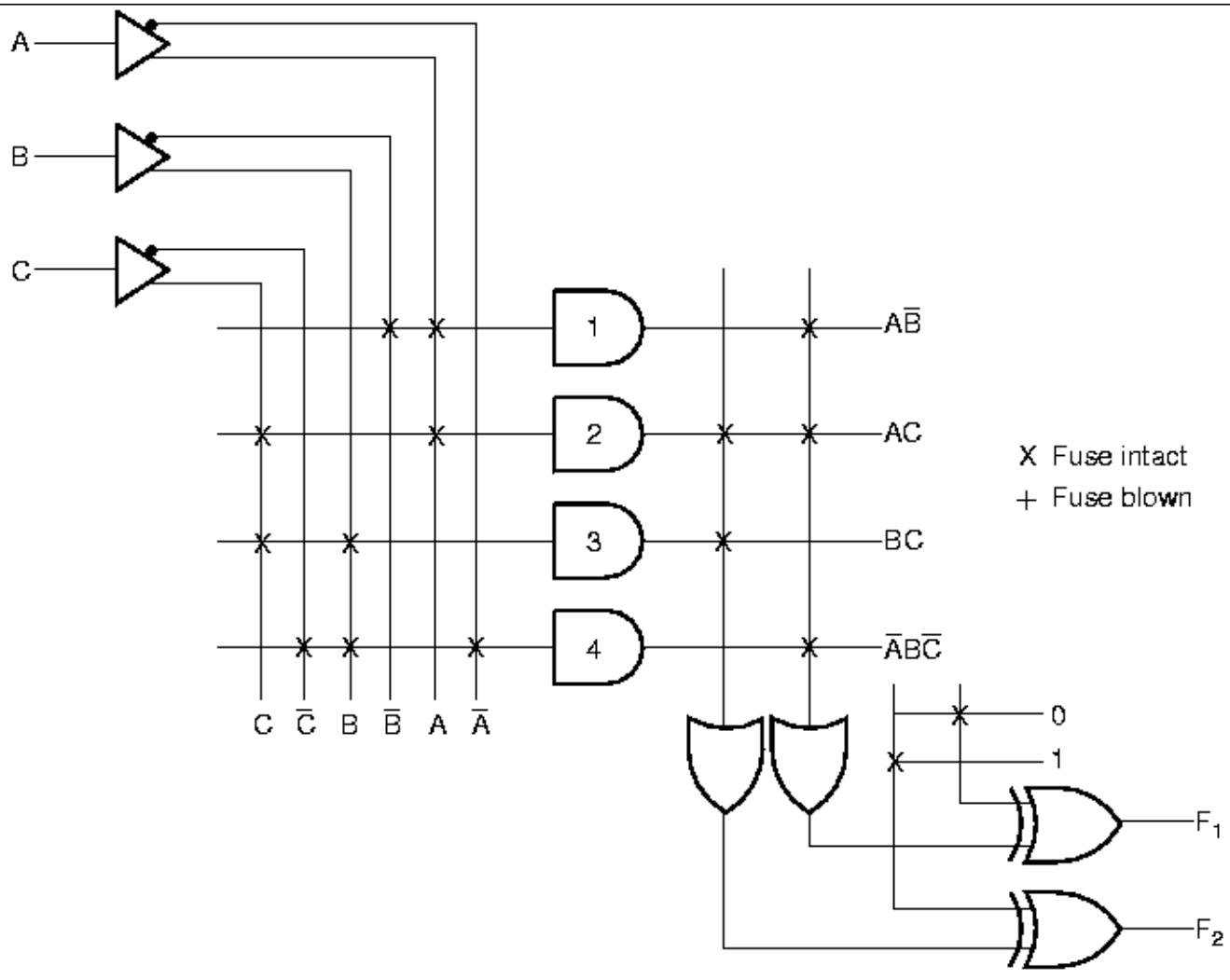
Behave like a ROM but has different structure

- Uses ANDs array instead of decoder to produce product terms of inputs
- Has programmable connections before ANDs, between ANDs and ORs, after ORs. That is $2nk + km + m$ fuses
- More flexible than ROM but more difficult to program
- Logic expressions for content information to be stored in PLA must be obtained fisrt, then minimized, and finally programmed into the PLA using a PLA program table
- PLA program table specifies product terms and sum terms of information that will be stored in PLA

Programming a PLA

PLA Program Table						
		Inputs			Outputs	
Term	Term#	A	B	C	F_1	F_2
$A\bar{B}$	1	1	0	_	1	_
AC	2	1	_	1	1	1
BC	3	_	1	1	_	1
$\bar{A}B\bar{C}$	4	0	1	0	1	_
					T	C

Corresponding PLA Implementation



Function Synthesis with PLA

Any set of functions $f_1(x_1, \ldots, x_n)$, ..., $f_m(x_1, \ldots, x_n)$ can be realized with a PLA

Example Implement $f_1(a, b, c) = \sum m(3, 5, 6, 7)$ and $f_2(a, b, c) = \sum m(0, 2, 4)$ with a PLA

First Simplify $f_1, \bar{f}_1, f_2, \bar{f}_2$, that is

$$f_1(a, b, c) = ab + ac + bc \qquad f_1, f_2 \rightarrow 5 \text{ terms}$$
$$\bar{f}_1(a, b, c) = \bar{a}\bar{b} + \bar{a}\bar{c} + \bar{b}\bar{c} \qquad f_1, \bar{f}_2 \rightarrow 4 \text{ terms}$$
$$f_2(a, b, c) = \bar{a}\bar{c} + \bar{b}\bar{c} \qquad \bar{f}_1, f_2 \rightarrow 3 \text{ terms}$$
$$\bar{f}_2(a, b, c) = ab + c \qquad \bar{f}_1, \bar{f}_2 \rightarrow 5 \text{ terms}$$

Second Select combination of functions that has less terms, that is

$$f_1 = \overline{\bar{f}_1} = \overline{\bar{a}\bar{b} + \bar{a}\bar{c} + \bar{b}\bar{c}}$$
$$f_2(a, b, c) = \bar{a}\bar{c} + \bar{b}\bar{c}$$

Third Construct a PLA program table from selected functions

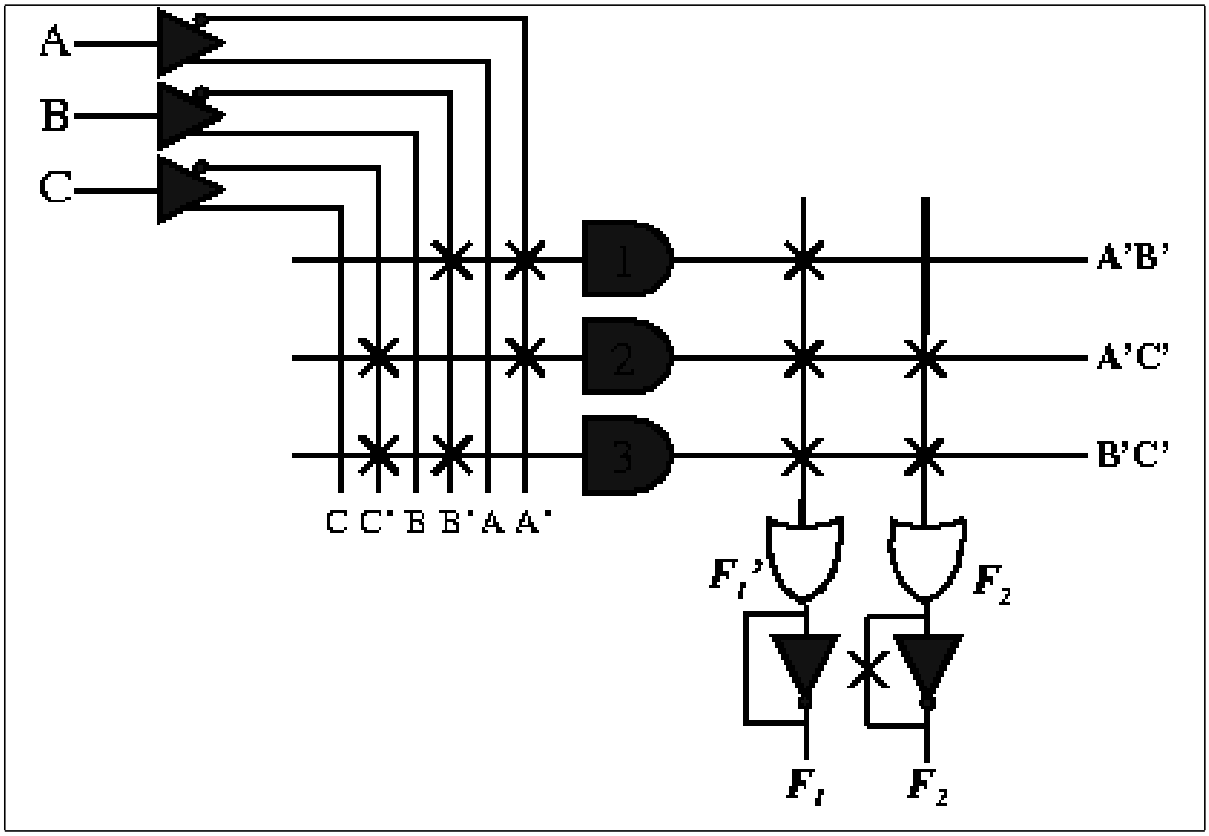
		Inputs			Outputs	
Term	Term#	a	b	c	f_1	f_2
$\bar{a}\bar{b}$	1	0	0	_	1	_
$\bar{a}\bar{c}$	2	0	_	0	1	1
$\bar{b}\bar{c}$	3	_	0	0	1	1
					C	T

Function Synthesis with PLA
(continued)

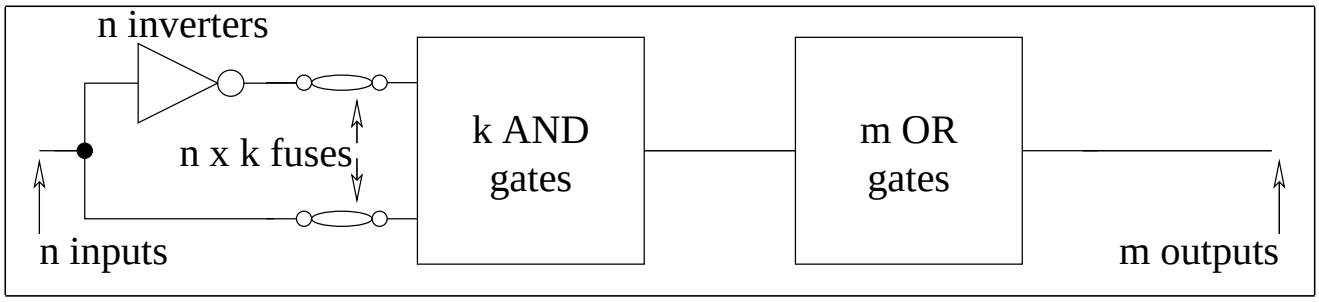
Third Construct a PLA program table from selected functions

Term	Term#	Inputs a	b	c	Outputs f_1	f_2
$\bar{a}\bar{b}$	1	0	0	_	1	_
$\bar{a}\bar{c}$	2	0	_	0	1	1
$\bar{b}\bar{c}$	3	_	0	0	1	1
					C	T

Fourth Construct PLA circuit from PLA program table



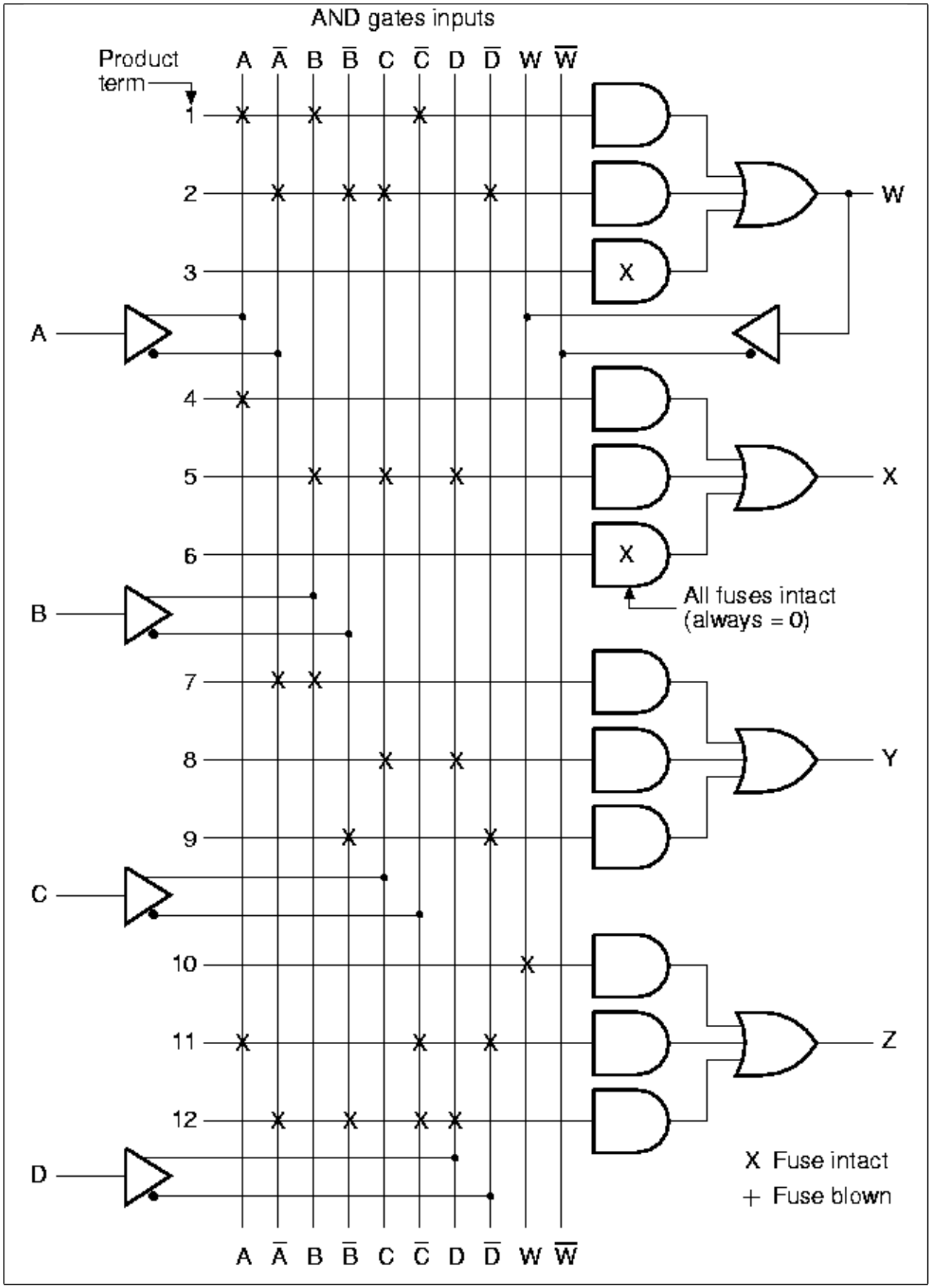
Programmable Array Logic



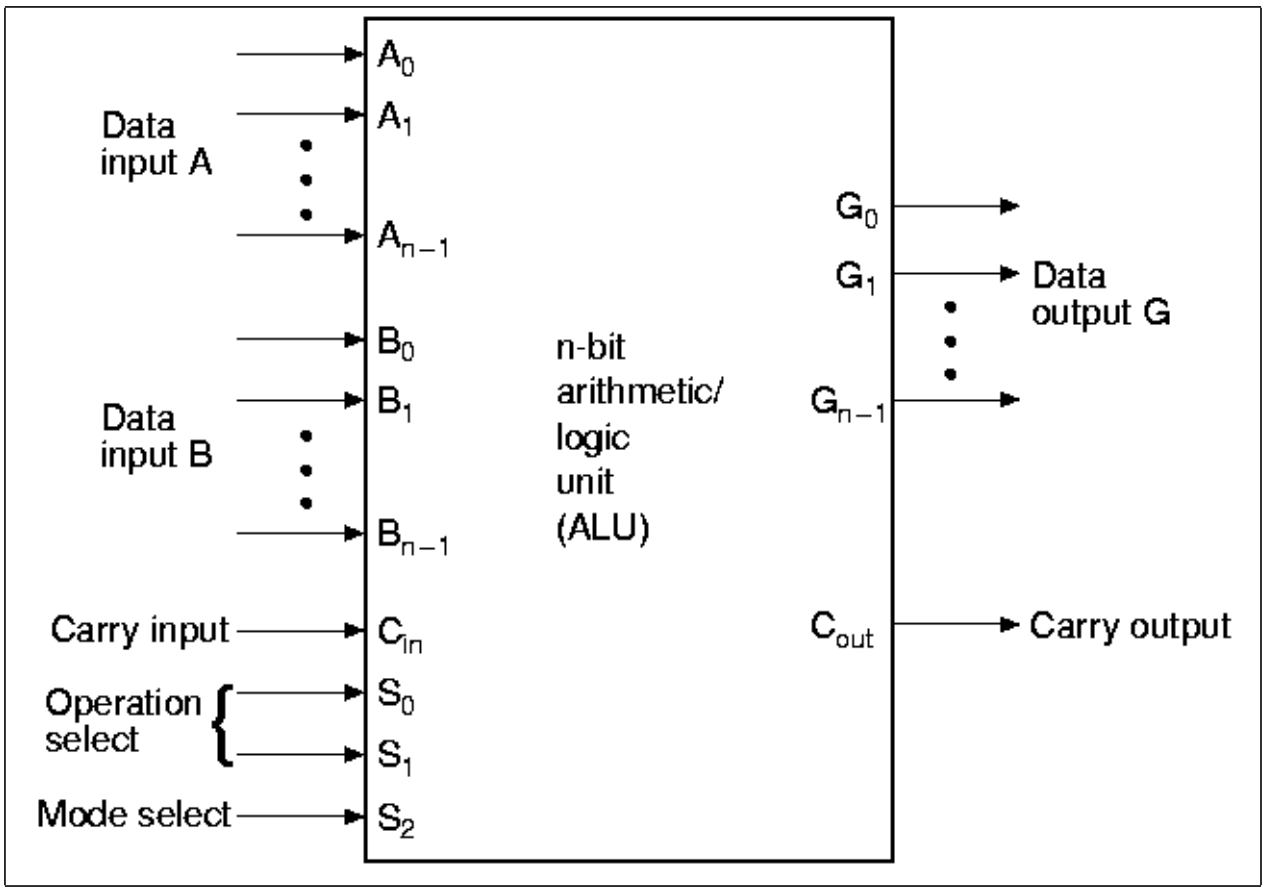
Similar to PLA

- Only the connection inputs to ANDs are programmable
- Easier to program than but not as flexible as PLA
- There are feedback connections
- Logic expressions for content information to be stored in PAL must be obtained fisrt, then minimized, and finally programmed into the PAL using a PAL program table
- PAL program table specifies only product terms of information that will be stored in PAL

Programming a PAL



Arithmetic-Logic Unit



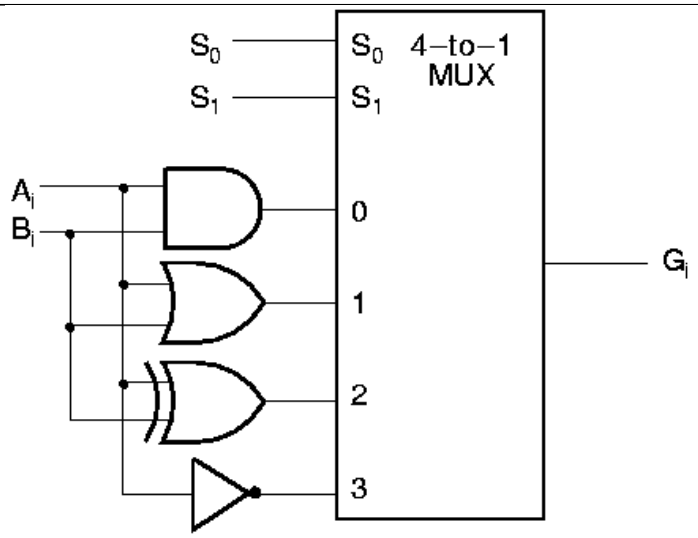
Essential element of the Central Processing Unit

Arithmetic and logic functions on binary words

- n-bit data inputs A and B
- n-bit data output $G = f(A, B)$
- Selection inputs S_0, S_1 select a function f
- Selection input S_2 select an operating mode (arithmetic or logic)

ALU
(continued)

Logic Circuit

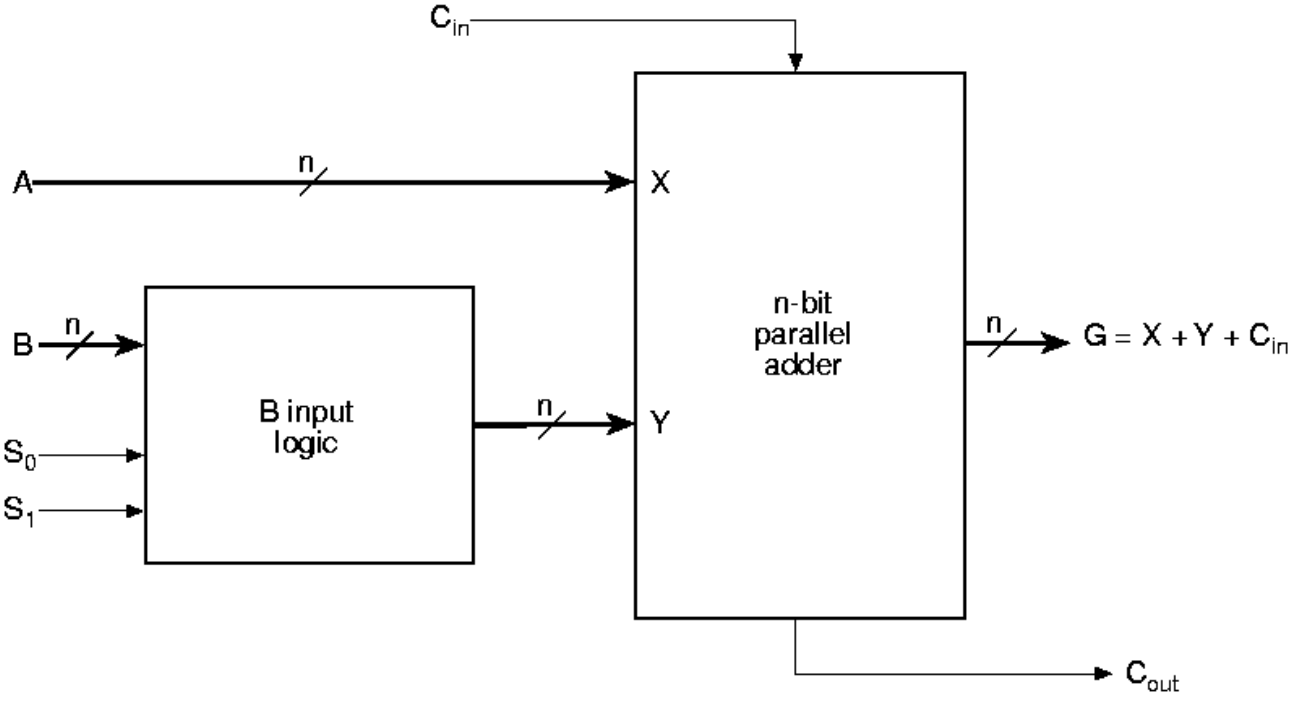


(a) Logic Diagram

S_1	S_0	Output	Operation
0	0	$G = A \wedge B$	AND
0	1	$G = A \vee B$	OR
1	0	$G = A \oplus B$	XOR
1	1	$G = \overline{A}$	NOT

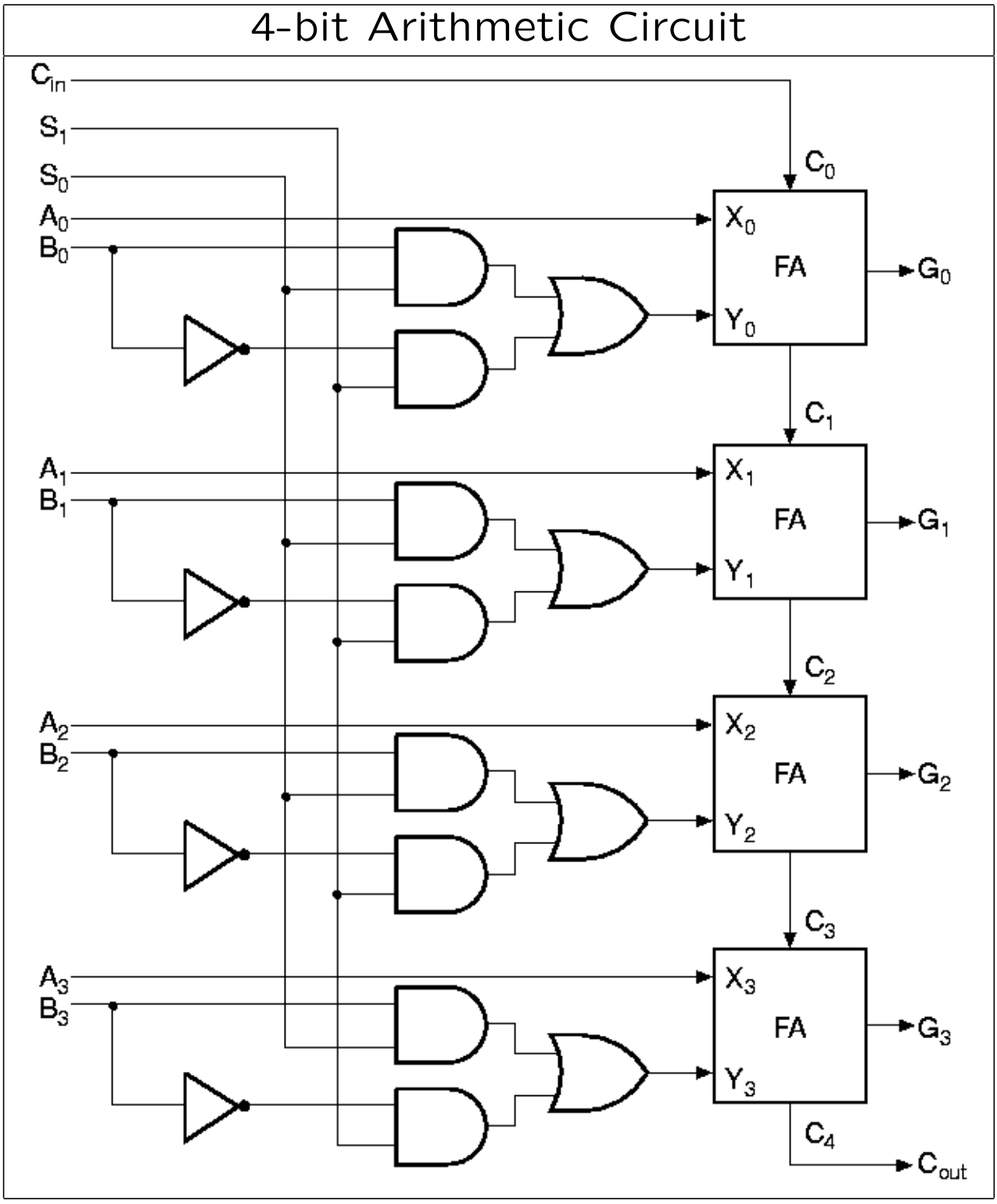
(b) Function Table

Arithmetic Circuit



ALU

(continued)



ALU
(continued)